Phase 5

Problem: Public Transport Efficiency Analysis

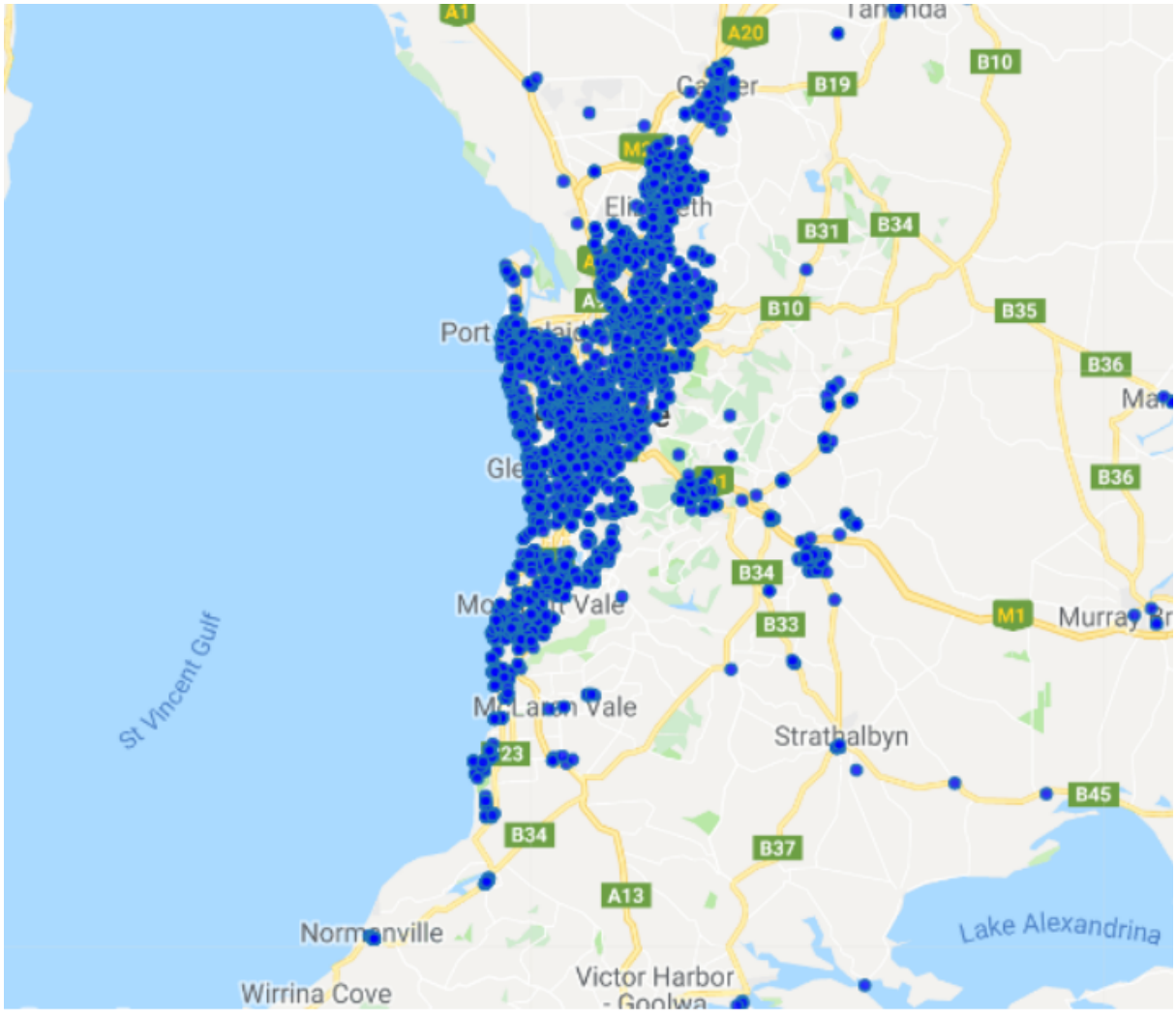
The first step is to import necessary libraries (Pandas, NumPy, Matplotlib, and Seaborn) for data analysis and visualization in Python.

```
%matplotlib inline
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import matplotlib.pyplot as plt
import datetime
import os
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import MinMaxScaler
import lightgbm as lgb
import xgboost as xgb
from sklearn.metrics import mean_squared_error
from math import sqrt
import warnings
warnings.filterwarnings('ignore')
print(os.listdir("../input/unisys/ptsboardingsummary"))
```

```
import plotly.plotly as py
import plotly.graph_objs as go
from plotly import tools
from plotly.offline import download_plotlyjs, init_notebook_mode, plot, iplot
from bubbly.bubbly import bubbleplot
init_notebook_mode(connected=True)

from bokeh.plotting import figure, save
from bokeh.io import output_file, output_notebook, show
from bokeh.models import ColumnDataSource, GMapOptions,HoverTool
from bokeh.plotting import gmap

import tensorflow as tf
from tensorflow.python.keras.models import Sequential
from tensorflow.python.keras.layers import Input, Dense, GRU,LSTM, Embedding
from tensorflow.python.keras.optimizers import RMSprop
from tensorflow.python.keras.callbacks import EarlyStopping, ModelCheckpoint, TensorBoard, ReduceLRO
nPlateau
```



The data fields in the given file are

- **TripID** Unique identity of trip
- **RouteID** Value representing public transport route
- **StopID** Unique identity of stop
- **StopName** Name of given stop
- **WeekBeginning** Date representing first day of any week
- **NumberOfBoarding** Count of all boarding's occurred at this stop for the named trip over the previous week.

Data Aggregation

```
data= pd.merge(data,out_geo,how='left',left_on = 'StopName',right_on = 'input_string')
```

```
data = pd.merge(data, route, how='left', left_on = 'RouteID', right_on = 'route_id')
```

Columns to keep for further analysis

```
col = ['TripID', 'RouteID', 'StopID', 'StopName', 'WeekBeginning','NumberOfBoardings','formatted_add
ress',
       'latitude', 'longitude','postcode','type','route_desc','dist_from_centre','holiday_label']
```

```
data = data[col]
```

Aggregate the Data According to Weeks and Stop names

- **NumberOfBoardings_sum** Number of Boardings within particular week for each Bus stop
- **NumberOfBoardings_count** Number of times data is recorded within week
- **NumberOfBoardings_max** Maximum number of boarding done at single time within week

```
# st_week_grp1 = pd.DataFrame(data.groupby(['StopName','WeekBeginning','type']).agg({'NumberOfBoarding
s': ['sum', 'count']})).reset_index()
grouped = data.groupby(['StopName','WeekBeginning','type']).agg({'NumberOfBoardings': ['sum', 'coun
t','max']})
grouped.columns = ["_".join(x) for x in grouped.columns.ravel()]
```

```
st_week_grp = pd.DataFrame(grouped).reset_index()
st_week_grp.shape
st_week_grp.head()
```

Gathering only the Stop Name which having all 54 weeks of Data

```
st_week_grp1 = pd.DataFrame(st_week_grp.groupby('StopName')['WeekBeginning'].count()).reset_index()
```

```
aa=list(st_week_grp1[st_week_grp1['WeekBeginning'] == 54]['StopName'])
```

```
bb = st_week_grp[st_week_grp['StopName'].isin(aa)]
```

```
## save the aggregate data
bb.to_csv('st_week_grp.csv', index=False)
```

Data Exploration

Total Having 1 Year of Data from date 2013-06-30 till 2014-07-06 in a Weekly interval based.

Having Total of 4165 Stops in the South Australian Metropolitan Area.

```
data.nunique()
```

```
data.shape
data.columns
data.head(3)
```

```
data.isnull().sum()
```

How Many different type of Unique Data in the dataset

```
data['WeekBeginning'].unique()
```

Data Visualization

```
##can assign the each chart to one axes at a time
fig,axrr=plt.subplots(3,2,figsize=(18,18))

data['NumberOfBoardings'].value_counts().sort_index().head(20).plot.bar(ax=axrr[0][0])
data['WeekBeginning'].value_counts().plot.area(ax=axrr[0][1])
data['RouteID'].value_counts().head(20).plot.bar(ax=axrr[1][0])
data['RouteID'].value_counts().tail(20).plot.bar(ax=axrr[1][1])
data['type'].value_counts().head(5).plot.bar(ax=axrr[2][0])
data['type'].value_counts().tail(10).plot.bar(ax=axrr[2][1])
```

Inferences:

- More than 40 lakhs times only single person board from the bus stop.
- There are average of 1.8 lakhs people travel every week by bus in adelaide metropolitan area.
- G10,B10,M44,H30 are the most busiest routes in the city while FX8,FX3,FX10,FX1,FX2 are the least.
- Most of the Bus stops are Street_Address Type while there are very few which are store or post office.

```
data['postcode'].value_counts().head(20).plot.bar()
```

```
# data['dist_from_centre'].nunique()
bb_grp = data.groupby(['dist_from_centre']).agg({'NumberOfBoardings': ['sum']}).reset_index()
bb_grp.columns = bb_grp.columns.get_level_values(0)
bb_grp.head()
bb_grp.columns
```

```
trace0 = go.Scatter(
    x = bb_grp['dist_from_centre'],
    y = bb_grp['NumberOfBoardings'],mode = 'lines+markers',name = 'X2 King William St')

data1 = [trace0]
layout = dict(title = 'Distance Vs Number of boarding',
              xaxis = dict(title = 'Distance from centre'),
              yaxis = dict(title = 'Number of Boardings'))
fig = dict(data=data1, layout=layout)
iplot(fig)
```

Inferences:

- As we move away from centre the number of Boarding decreases
- There are cluster of bus stops near to the main Adelaide city as oppose to outside.so that's why most of boardings are near to center.

Using Bokeh

Plot the Bus stop on the Google Map using the latitude and longitude of the bus stop address

```
lat = out_geo['latitude'].tolist()
long = out_geo['longitude'].tolist()
nam = out_geo['input_string'].tolist()
```

```
map_options = GMapOptions(lat=-34.96, lng=138.592, map_type="roadmap", zoom=9)
key = open('../input/geolockey/api_key.txt').read()
p = gmap(key, map_options, title="Adelaide South Australia")
source = ColumnDataSource(data=dict(lat=lat,lon=long,nam=nam))

p.circle(x="lon", y="lat", size=5, fill_color="blue", fill_alpha=0.8, source=source)
TOOLTIPS = [("Place", "@nam")]
p.add_tools( HoverTool(tooltips=TOOLTIPS))
output_notebook()
show(p)
```

Inferences:

- It has Geospatial coverage Area from Lat: 34.3862 to -35.3655 and Lon: 138.4126 to 139.1089. Which is Total 152 KM long Area from Daniel Road to Mosquito Creek Road on one side and Total 162 KM Stretch from Truro to Myponga Beach on the other side.
- There are cluster of bus stops near to the main Adelaide city as oppose to outside.

Plot using Plotly

```
## for finding highest number of Boarding Bus stops
bb_grp = bb.groupby(['StopName']).agg({'NumberOfBoardings_sum': ['sum']}).reset_index()['NumberOfBoa
rdings_sum'].sort_values('sum')
bb_grp[1000:1005]
bb.groupby(['StopName']).agg({'NumberOfBoardings_sum': ['sum']}).reset_index().iloc[[2325,1528,546,1
043,1905]]
# bb_grp.iloc[[3054]]
```

```
source_1 = bb[bb['StopName'] == 'X2 King William St'].reset_index(drop = True)
source_2 = bb[bb['StopName'] == 'E1 Currie St'].reset_index(drop = True)
source_3 = bb[bb['StopName'] == 'I2 North Tce'].reset_index(drop = True)
source_4 = bb[bb['StopName'] == 'F2 Grenfell St'].reset_index(drop = True)
source_5 = bb[bb['StopName'] == 'D1 King William St'].reset_index(drop = True)
```

```
trace0 = go.Scatter(
    x = source_1['WeekBeginning'],
    y = source_1['NumberOfBoardings_sum'],mode = 'lines+markers',name = 'X2 King William St')
trace1 = go.Scatter(
    x = source_2['WeekBeginning'],
    y = source_2['NumberOfBoardings_sum'],mode = 'lines+markers',name = 'E1 Currie St')
trace2 = go.Scatter(
    x = source_3['WeekBeginning'],
    y = source_3['NumberOfBoardings_sum'],mode = 'lines+markers',name = 'I2 North Tce')
trace3 = go.Scatter(
    x = source_4['WeekBeginning'],
    y = source_4['NumberOfBoardings_sum'],mode = 'lines+markers',name = 'F2 Grenfell St')
trace4 = go.Scatter(
    x = source_5['WeekBeginning'],
    y = source_5['NumberOfBoardings_sum'],mode = 'lines+markers',name = 'D1 King William St')
```

```
data = [trace0,trace1,trace2,trace3,trace4]
layout = dict(title = 'Weekly Boarding Total',
              xaxis = dict(title = 'Week Number'),
              yaxis = dict(title = 'Number of Boardings'),
              shapes = [{# Holidays Record: 2013-09-01
'type': 'line','x0': '2013-09-01','y0': 0,'x1': '2013-09-02','y1': 18000,'line': {
        'color': 'rgb(55, 128, 191)','width': 1,'dash': 'dashdot'},},
              {# 2013-10-07
'type': 'line','x0': '2013-10-07','y0': 0,'x1': '2013-10-07','y1': 18000,'line': {
        'color': 'rgb(55, 128, 191)','width': 1,'dash': 'dashdot'},},
              {# 2013-12-25
'type': 'line','x0': '2013-12-25','y0': 0,'x1': '2013-12-26','y1': 18000,'line': {
        'color': 'rgb(55, 128, 191)','width': 3,'dash': 'dashdot'},},
              {# 2014-01-27
'type': 'line','x0': '2014-01-27','y0': 0,'x1': '2014-01-28','y1': 18000,'line': {
        'color': 'rgb(55, 128, 191)','width': 1,'dash': 'dashdot'},},
              {# 2014-03-10
'type': 'line','x0': '2014-03-10','y0': 0,'x1': '2014-03-11','y1': 18000,'line': {
        'color': 'rgb(55, 128, 191)','width': 1,'dash': 'dashdot'},},
              {# 2014-04-18
'type': 'line','x0': '2014-04-18','y0': 0,'x1': '2014-04-19','y1': 18000,'line': {
        'color': 'rgb(55, 128, 191)','width': 3,'dash': 'dashdot'},},
              {# 2014-06-09
'type': 'line','x0': '2014-06-09','y0': 0,'x1': '2014-06-10','y1': 18000,'line': {
        'color': 'rgb(55, 128, 191)','width': 1,'dash': 'dashdot'},},])
fig = dict(data=data, layout=layout)
iplot(fig)
```

Inferences:

- X2 King William St and stop near to that are the most busiest stops in the city. which having number of boardings per week more than 10k.
- Vertical lines are the indicator of holidays which came within that week.
- Whenever there is any Public holiday that week period have less than average number of people travelled from bus.

```
source_6 = bb[bb['StopName'] == '57A Hancock Rd'].reset_index(drop = True)
source_7 = bb[bb['StopName'] == '37 Muriel Dr'].reset_index(drop = True)
source_8 = bb[bb['StopName'] == '18B Springbank Rd'].reset_index(drop = True)
source_9 = bb[bb['StopName'] == '27E Sir Ross Smith Av'].reset_index(drop = True)
source_10 = bb[bb['StopName'] == '46A Baldock Rd'].reset_index(drop = True)
```

Inferences:

- Same decreasing affect of Holidays on number of people travelling through bus can be seen in other city bus stops also.
- The width of vertical blue line shows the number of holidays come within that week period.
- Two thickest blue lines shows Christmas and New year period while other one was easter & Good friday period. on both the occassion number of public holidays within week period was 3.

Plot Using Bubbly

```
figure = bubbleplot(dataset=bb1, x_column='NumberOfBoardings_sum', y_column='NumberOfBoardings_coun
t',
    bubble_column='StopName', time_column='WeekBeginning', size_column='NumberOfBoardings_max',
    color_column='type',
    x_title="Total Boardings", y_title="Frequency Of Boardings",show_slider=True,
    title='Adelaide Weekly Bus Transport Summary 2D',x_logscale=True, scale_bubble=2,height=650)

iplot(figure, config={'scrollzoom': True})
```

In the graph above, the size corresponds to the maximum number of people board at single time and the Total boardings and Frequency of boardings with stop name can be seen by hovering over the cursor on the bubbles.

The animated bubble charts convey a great deal of information since they can accomodate upto seven variables in total, namely:

- X-axis (Total Boardings per week)
- Y-axis (Frequency of Bus Boarding)
- Bubbles (Bus stop name)
- Time (in week period)
- Size of bubbles (maximum number of people board at single time)
- Color of bubbles (Type of Bus stop)

```
figure = bubbleplot(dataset=bb1[bb1['StopName'].isin(bb1['StopName'].unique()[:30])], x_column='Numb
erOfBoardings_sum', y_column='NumberOfBoardings_count',
    bubble_column='StopName', time_column='WeekBeginning', size_column='NumberOfBoardings_max',
    color_column='type',
    x_title="Total Boardings", y_title="Frequency Of Boardings",show_slider=False,
    title='Adelaide Weekly Bus Transport Summary 2D',x_logscale=True, scale_bubble=2,height=650)

iplot(figure, config={'scrollzoom': True})
```

```
figure = bubbleplot(dataset=bb1, x_column='NumberOfBoardings_sum', y_column='NumberOfBoardings_coun
t',
    bubble_column='StopName', time_column='WeekBeginning', z_column='NumberOfBoardings_max',
    color_column='type',show_slider=False,
    x_title="Total Boardings", y_title="Frequency Of Boardings", z_title="Maximum Boardings",
    title='Adelaide Weekly Bus Transport Summary 3D', x_logscale=True, z_logscale=True,y_logscale=Tr
ue,
    scale_bubble=0.8, marker_opacity=0.8, height=700)

iplot(figure, config={'scrollzoom': True})
```

Inferences:

- Total Boardings are directly propotional to the frequency of bus boarding.
- In 3D Plot we can see the cluster of address type.

Propositions

Rate of change in the traffic pattern in all different bus stops.

```
d=[]
for i in bb['StopName'].unique():
    d.append({'StopName': i,'Boarding_sum':np.sum(bb[bb['StopName'] == i]['NumberOfBoardings_sum'].p
ct_change())/54,
              'Boarding_count':np.sum(bb[bb['StopName'] == i]['NumberOfBoardings_count'].pct_change
())/54,
              'Boarding_max':np.sum(bb[bb['StopName'] == i]['NumberOfBoardings_max'].pct_change())/5
4})
pct_chng = pd.DataFrame(d)
```

```
#pct_chng.head()
pct_chng['Boarding_sum'].nlargest(5)
pct_chng['Boarding_sum'].nsmallest(5)
pct_chng[pct_chng['Boarding_sum']<0].shape
pct_chng.iloc[[3110,2134,214,1538,1290]]
```

Inferences:

- These 5 stops W Grote St,52 Taylors Rd,13 Tutt Av,37A Longwood Rd,32A Frederick Rd having the largest percent increase.
- There are 27 Bus stops where number of boardings have decreased.
- The number of busses can be found by taking the number of boarding divided by bus capacity which can take as 50.

Predictive Modeling

Get info like RouteID,latitude,longitude,postcode,dist_from_centre & holiday_label 6 features from the main dataset

```
bb1 = pd.merge(bb, out_geo, how='left', left_on = 'StopName', right_on = 'input_string')
```

```
bb1['holiday_label'] = bb1['WeekBeginning'].apply (lambda row: holiday_label(row))
```

```
##Final 11 features have been used for the forecastng.
cols = ['StopName','WeekBeginning','type_x','NumberOfBoardings_sum','NumberOfBoardings_count','Numbe
rOfBoardings_max','latitude','longitude','postcode','dist_from_centre','holiday_label']
bb1=bb1[cols]
bb1.shape
bb1.head()
```

```
##Replace all Nan by Mode
for i in bb1.columns:
    bb1[i].fillna(bb1[i].mode()[0], inplace=True)
bb1[["postcode", "holiday_label"]] = bb1[["postcode", "holiday_label"]].apply(pd.to_numeric)
```

```
tr_col = ['StopName', 'WeekBeginning', 'type_x', 'latitude',
       'longitude', 'postcode', 'dist_from_centre', 'holiday_label']
train_sum_y = train[['StopName','NumberOfBoardings_sum']]
train_count_y = train[['StopName','NumberOfBoardings_count']]
train_max_y = train[['StopName','NumberOfBoardings_max']]
train_x = train[tr_col]
test_x = test[tr_col]

test_sum_y = test[['StopName','NumberOfBoardings_sum']]
test_count_y = test[['StopName','NumberOfBoardings_count']]
test_max_y = test[['StopName','NumberOfBoardings_max']]
```

Modeling using regression models.

1. lightGbm Regressor
2. Gru

Using LightGbm

```
from sklearn.ensemble import RandomForestRegressor
# model = lgb.LGBMRegressor()
model = RandomForestRegressor(n_estimators=700, min_samples_leaf=3, max_features=0.5,n_jobs=-1)
# model = lgb.LGBMRegressor(max_depth=10,learning_rate=0.0227,n_estimators=195,num_leaves=11,reg_alpha
=1.5764,reg_lambda=0.0478,subsample=0.7776,colsample_bytree=0.7761)
model.fit(train_x.values,train_sum_y['NumberOfBoardings_sum'].values)
preds = model.predict(test_x.values)
```

```
rms = sqrt(mean_squared_error(test_sum_y['NumberOfBoardings_sum'].values, preds))
rms
```

```
test_sum_y.values[:15]
preds[:15]
```

```
fig, ax = plt.subplots(figsize=(6,10))
lgb.plot_importance(model, max_num_features=50, height=0.8, ax=ax)
ax.grid(False)
plt.title("LightGBM - Feature Importance", fontsize=15)
plt.show()
```

```
plt.figure(figsize=(15,5))
plt.plot(test_sum_y['NumberOfBoardings_sum'].values, label='true')
plt.plot(preds, label='pred')
plt.ylabel("Total Number of Boarding")
plt.xlabel("Index")
plt.title("Comparison Between Prediction & True Values")
plt.legend()
plt.show()
```

Conclusion:

Thus the model was trained for the Australian Traffic dataset using python libraries like pandas, numpy, seaborn and matplotlib.